

|  |
| --- |
| **Stochastic Simulation - TP5** |

Jiarui XIE

Semester: 2018/2019

Superviser: David Hill

Catalog

[I. Introduction 1](#_Toc529611338)

[II. Rabbit population growth 1](#_Toc529611339)

[A. Flow-Process Diagram 1](#_Toc529611340)

[B. Description of the Parameters 1](#_Toc529611341)

[C. Performances 2](#_Toc529611342)

[D. Simulate over Year 2](#_Toc529611343)

[E. Simulate on Different Parameters 3](#_Toc529611344)

[F. Confidence Interval at 95% 3](#_Toc529611345)

[III. CONCLUSION 4](#_Toc529611346)

# Introduction

In this exercise, we will simulate the growth of a rabbit population in the absence of predators at the same time we will take into account the birth and death of rabbits.

First of all, we suppose as long as there is a male rabbit all of the female rabbits will be fertilized every month. If there is not male rabbit we will wait the pups get matured until reach the end of the simulation time.

Secondly, the initialization is there are two matured couple rabbits.

# multi-agent simulation

## Implement a game of life in “text” mode

We simulate the game of life on a 5x5 cell space with the glider’s initial state. It will end at the 8th iteration without torus neighborhood and the loop period is 20 with a torus universe. The running result is the same as the result on the website.

Like Figure1.1.

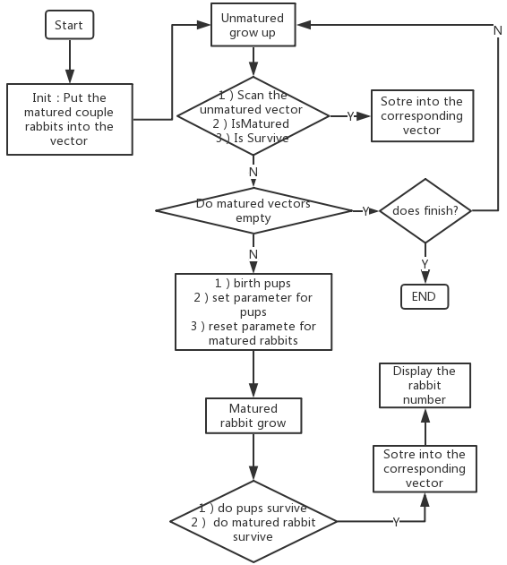


Figure1.1 Flow-process Diagram of this lab code

## Hashlife Algorithm

Hashlife is a memoized algorithm for computing the long-term fate of a given starting configuration in Conway's Game of Life and related cellular automata, much more quickly than would be possible using alternative algorithms that simulate each time step of each cell of the automaton.

HashLife uses Bill Gosper's hashlife algorithm to achieve remarkable speeds when generating patterns that have a lot of regularity in time and/or space.

Roughly speaking, the idea of the algorithm is to store subpatterns in a hash table so that the results of their evolution don't have to be recomputed if they arise again at another place or time.

HashLife provides a means of evolving repetitive patterns millions (or even billions or trillions) of generations further than normal Life algorithms such as QuickLife can manage in a reasonable amount of time. It is not, generally, suitable for showing a continuous display of the evolution of a pattern, because it works asynchronously — at any given moment it will usually have evolved different parts of the pattern through different numbers of generations.

How does the hashlife algorithm go on forever in Golly? The basic answer is that you don't merely run the algorithm on the existing nodes, you also use new shifted nodes to get the next generation.

An algorithm accelerating the calculation of cellular automata in a phenomenal way.

Hashlife is able to calculate thousands of generations per second of "Games of Life" with 10 ^ 50 cells on a regular PC with 10 ^ 9 bytes of memory!

Figure1.2 histogram of the number of litters per female.

Figure1.3 histogram of the number of pups of a female per litter.

Figure1.4 histogram of the time needed to mature of a pup

## Memoization

We tested the performance of our simulator based on the years with initialization is two couples on my local computer. The results for 5、6、7、8 and 9-year like Figure1.5. Obversely, the more initialization couples the more time it needs.

Figure1.5 time needed to simulate on different years

## Simulate over Year

We first simulated the reproduction of the population over different lengths of time. The graph in Figure1.6 shows the results obtained for 5、6、7、8 and 9-year simulations.

It appears on these graphs that whatever the simulate time the population obtained follows an exponential growth. This result is perfectly logical: the more adults there are, the more we get rabbits.

Figure1.6 Evolution of the size of the population

## Simulate on Different Parameters

We get the number of rabbits on different parameters, they are initialization number is two couples with the ratio of matured survived is 50% with the ratio of unmatured survived is 20%（2-20-50）、initialization number is two couples with the ratio of matured survived is 80% with the ratio of unmatured survived is 90%（2-80-90）、initialization number is four couples with the ratio of matured survived is 50% with the ratio of unmatured survived is 20%（4-20-50）、initialization number is two couple with the ratio of matured survived is 80% with the ratio of unmatured survived is 90%（4-80-90）.

Figure1.7 simulation based on different parameter

## Confidence Interval at 95%

We used the formulation in Lab3 to calculate the confidence interval at 95%. We used *init\_genrand((unsigned long)time(NULL));* to get independent random streams and repeat ten times. The initialization number is two couples with the ratio of matured survived is 50% with the ratio of unmatured survived is 20%, simulated for five years.

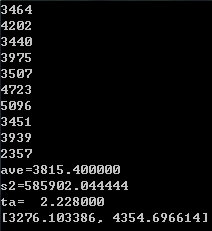


Figure1.8 confidence interval at 95%

# CONCLUSION

We have been able to establish a correct general model which seems to correspond to the results of our comrades. However, to be faithful to a real case, it lacks the consideration of many environmental parameters (terrain, nature, quantity of food, predators, diseases, and competitors...) and some intrinsic parameters, which can vary according to the species. Thus, our model can provide results far removed from actual observed results.